Web Development I

Les 5: CSS basis deel 2

HO GENT

Inhoud

- Inleiding Cascading - Inheritance
- CSS verwerking door browsers: cascading - inheritance
- Cascading
  - Origin, importance
  - Specificity
  - Source order
- Inheritance
  - Overerfbare eigenschappen
  - Wat met niet overerfbare eigenschappen?
- Developers Tools - CSS verwerking door browsers
- CSS values and units
- Browser reset/normalize.css
- Web fonts

De ‘Cascade’ - inleiding

Cascading Style Sheets

- Eén van de fundamentele ontwerpprincipes van CSS is ‘cascading’. Vandaar de naam Cascading Style Sheets.
  Soms zullen er meerdere CSS stijlen van toepassing zijn op eenzelfde element. In dat geval bepaalt de ‘Cascade’ welke CSS stijl er zal toegepast worden.

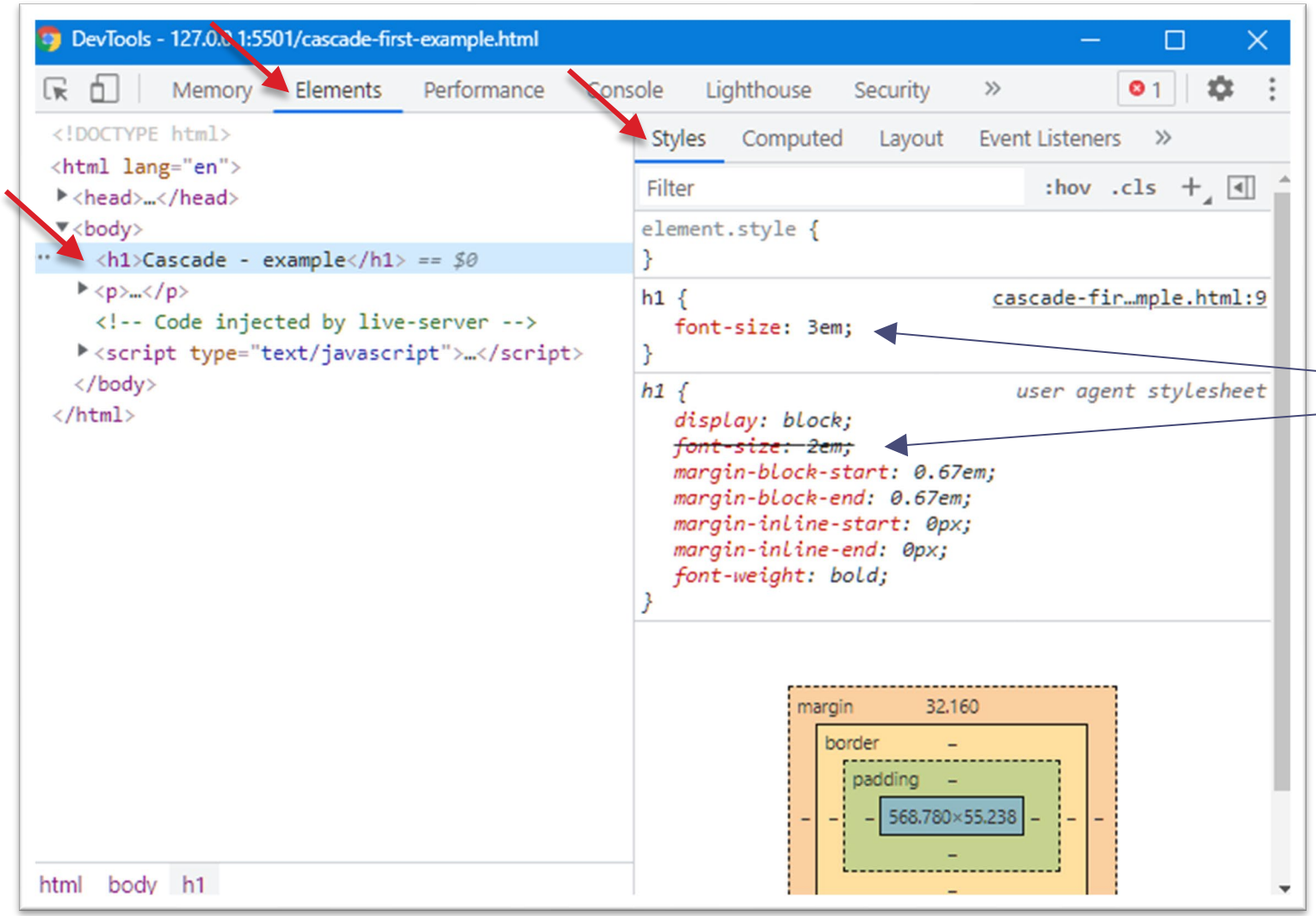
Inleidend voorbeeld: de ‘Cascade’ – origin

- Open **01-inleiding-cascade-inheritance/cascade-first-example.html**

CSS: user agent stylesheet/author stylesheet

- We weten al dat veel elementen standaard al een opmaak hebben, zonder dat wij hiervoor een CSS stijl moeten instellen. Zo hebben h1-elementen standaard een groter lettertype. Deze opmaak is afkomstig van de ‘browser stylesheet’. In technische documenten meestal de ‘**user agent stylesheet**’ genoemd.
- Als we bijgevolg zelf ook een CSS stijl instellen voor het h1-element, dan zijn er twee CSS stijlen ingesteld voor het h1-element. Eén in de browser stylesheet en in onze stylesheet (in technische documenten meestal de ‘**author stylesheet**’ genoemd). De vraag is dan: welke zal er toegepast worden. Met andere woorden welke heeft de hoogste prioriteit. In dit geval voel je aan dat jouw CSS stijl deze van de browser zal overschrijven.

Developer Tools (cascade-first-example.html)



Met de Developer Tools kunnen we kijken welke stijlen er ingesteld zijn voor een bepaald HTML-element. En we kunnen ook zien welke stijlen er overschreven worden.

de `font-size: 2em`, ingesteld voor het h1-element, in de user agent stylesheet wordt overschreven door `font-size: 3em` ingesteld in de author stylesheet.

HO GENT

Inleidend voorbeeld: de ‘Cascade’ – specificity

- Open **01-inleiding-cascade-inheritance/cascade-second-example.html**

cascade-second-example.html - specificity

- Soms zullen we ook zelf twee CSS stijlen creëren die van toepassing zijn op eenzelfde element.

CSS

```css
/* elementen met een class extlink
krijgen een rode kleur */
.extlink {
  color: red;
}
/* alle a-elementen krijgen een
groene kleur  */
a {
  color: green;
}
```

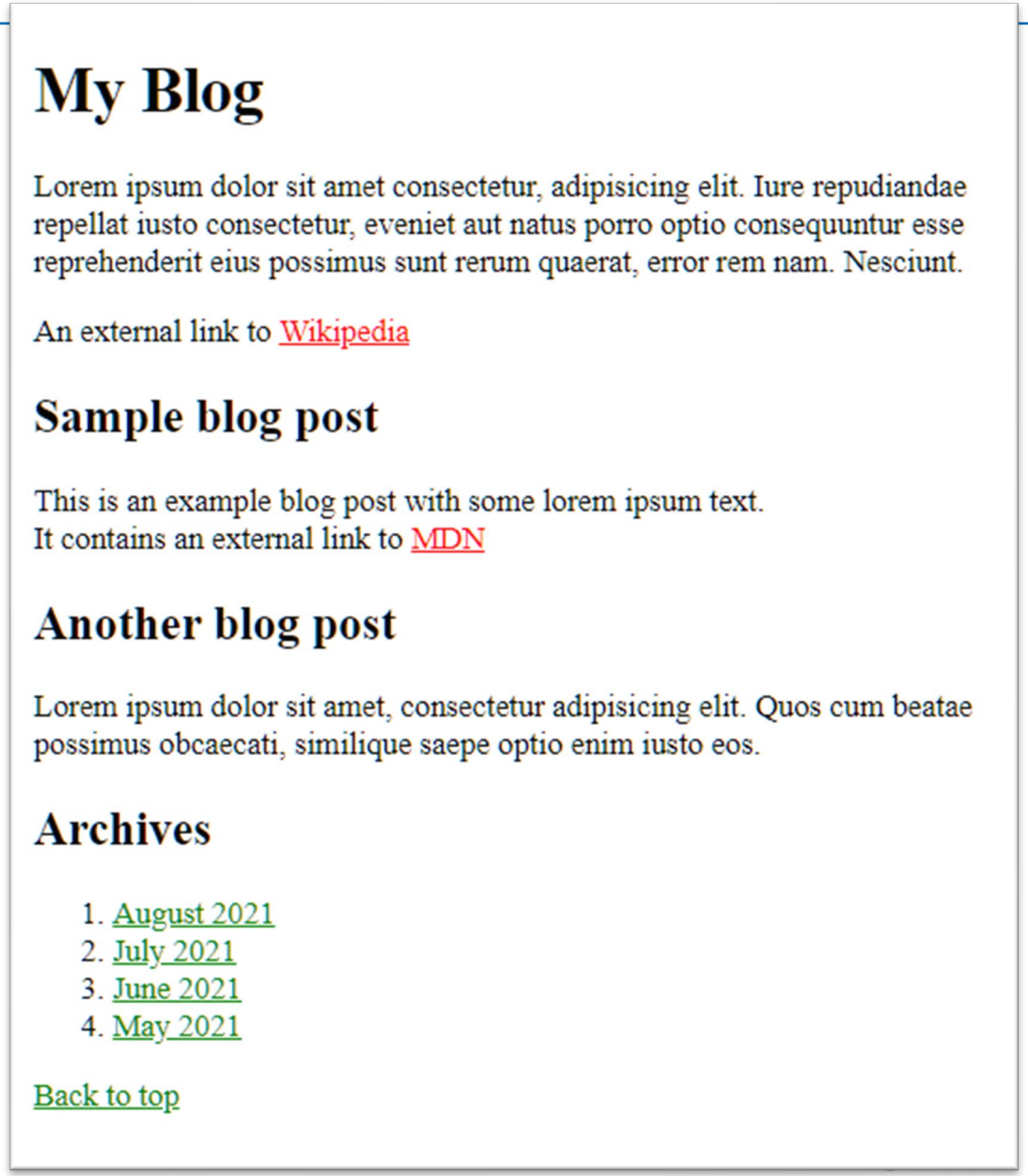
HTML

```html
<p>
  An external link to
  <a href=https://en.wikipedia.org/
     class="extlink">Wikipedia</a>
</p>
```

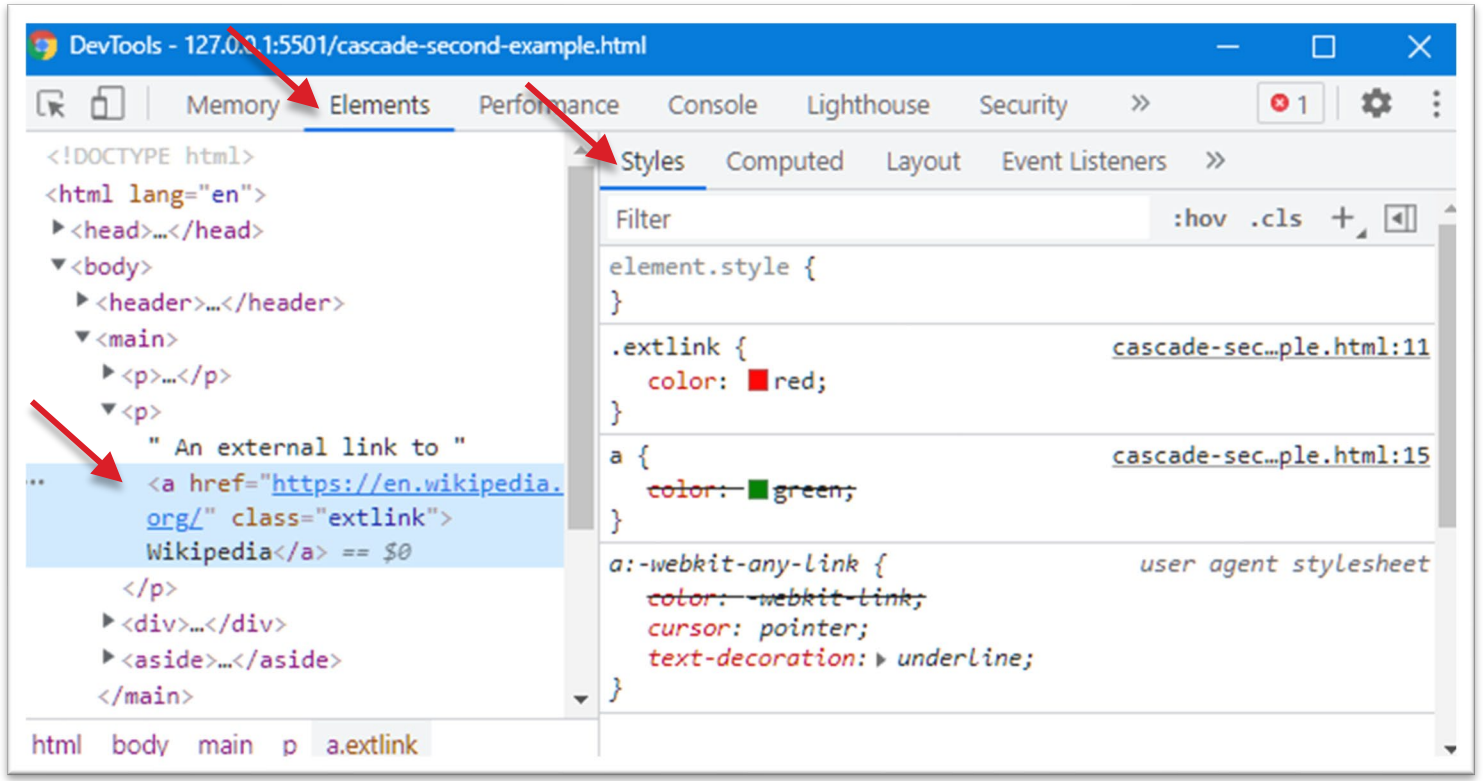
Welke kleur zal de wikipedia-link krijgen?

cascade-second-example.html - specificity

- De wikipedia-link zal een rode kleur krijgen omdat de class selector specifieker is dan de type selector. (Technischer gezegd: de class selector heeft een hogere ‘specificity’ dan de type selector, zie verderop).



Developer Tools (cascade-second-example.html)



- De `color` wordt 3 keer ingesteld. Eén keer in de user agent stylesheet en twee keer in de author stylesheet.
- De kleur **-webkit-link** wordt overschreven door **groen**.
  En de groene kleur wordt overschreven door rood.

HO GENT

Inheritance - inleiding

Inleidend voorbeeld: inheritance

- Open **01-inleiding-cascade-inheritance/inheritance-example.html**

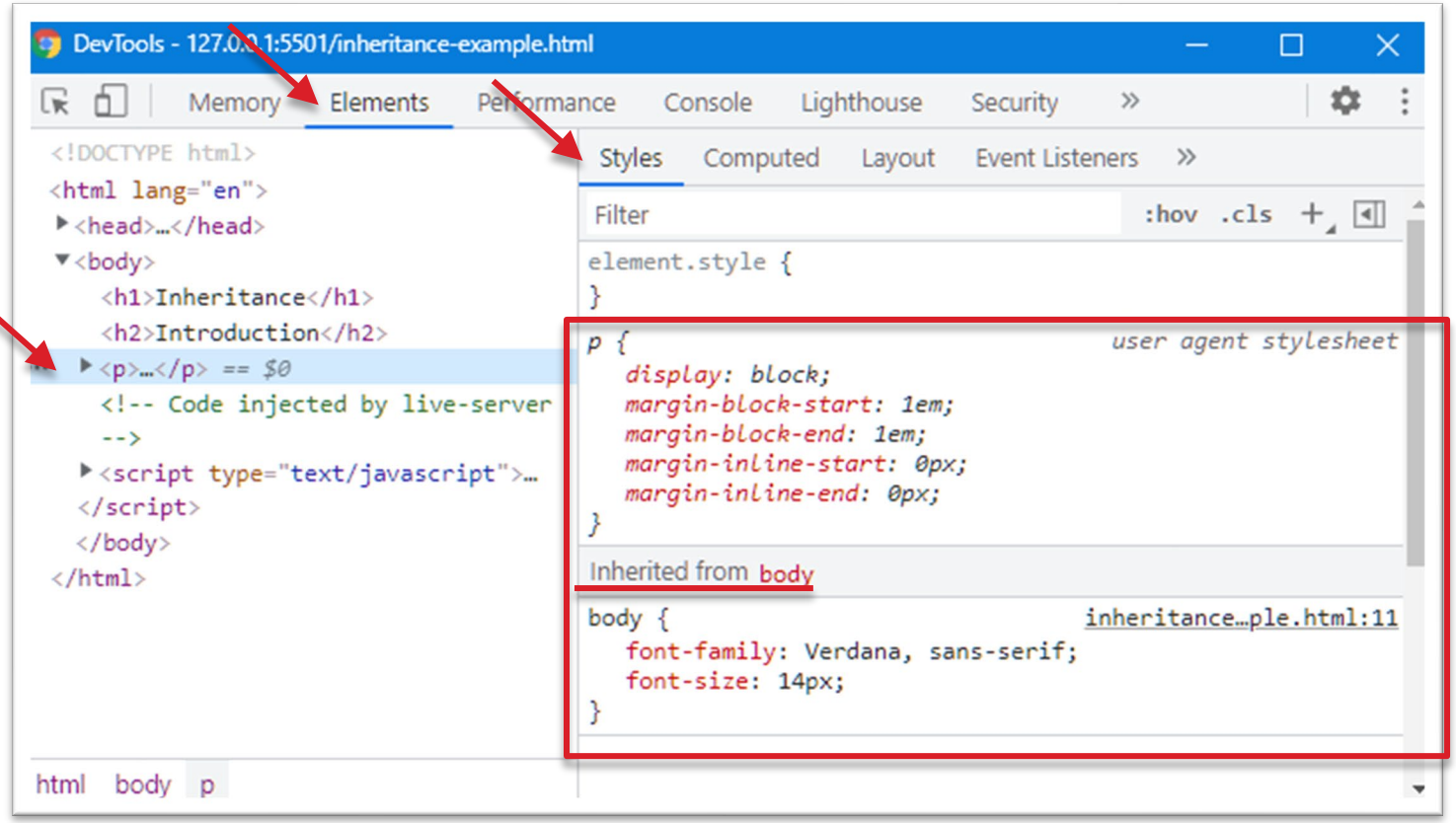
Inheritance - inleiding

- Als we een article-element en als zijn ‘descendants’ dezelfde tekstkleur willen geven, dan moeten we niet voor elk van de ‘descendants’ de tekstkleur instellen. Het volstaat om de kleur in te stellen op het article-element zelf. De ‘descendants’ van het article-element erven dan de kleur over.
- Niet alle CSS-eigenschappen worden overgeërfd. CSS-eigenschappen die overgeërfd worden zijn vnl. tekst gerelateerde CSS-eigenschappen.
- Daar het body-element alle elementen van onze webpagina bevat zullen we dus meestal font-family en font-size instellen op het body-element. Alle andere elementen zullen dan deze properties overerven.
  En alle tekst zal dezelfde font-family en font-size hebben, tenzij we voor een tekst expliciet een andere font-family of font-size instellen.

inheritance-example.html

```
/* globale tekst gerelateerde CSS properties
stellen we meestal in op het body element */
body {
  font-family: Verdana, sans-serif;
  font-size: 14px;
}

/* voor de heading-elementen stellen we
expliciet een andere font-family in  */
h1, h2 {
  font-family: Georgia, serif;
}
```

Developer Tools (inheritance-example.html)



Voor het p-element wordt er niet expliciet een `font-family` en een `font-size` ingesteld.

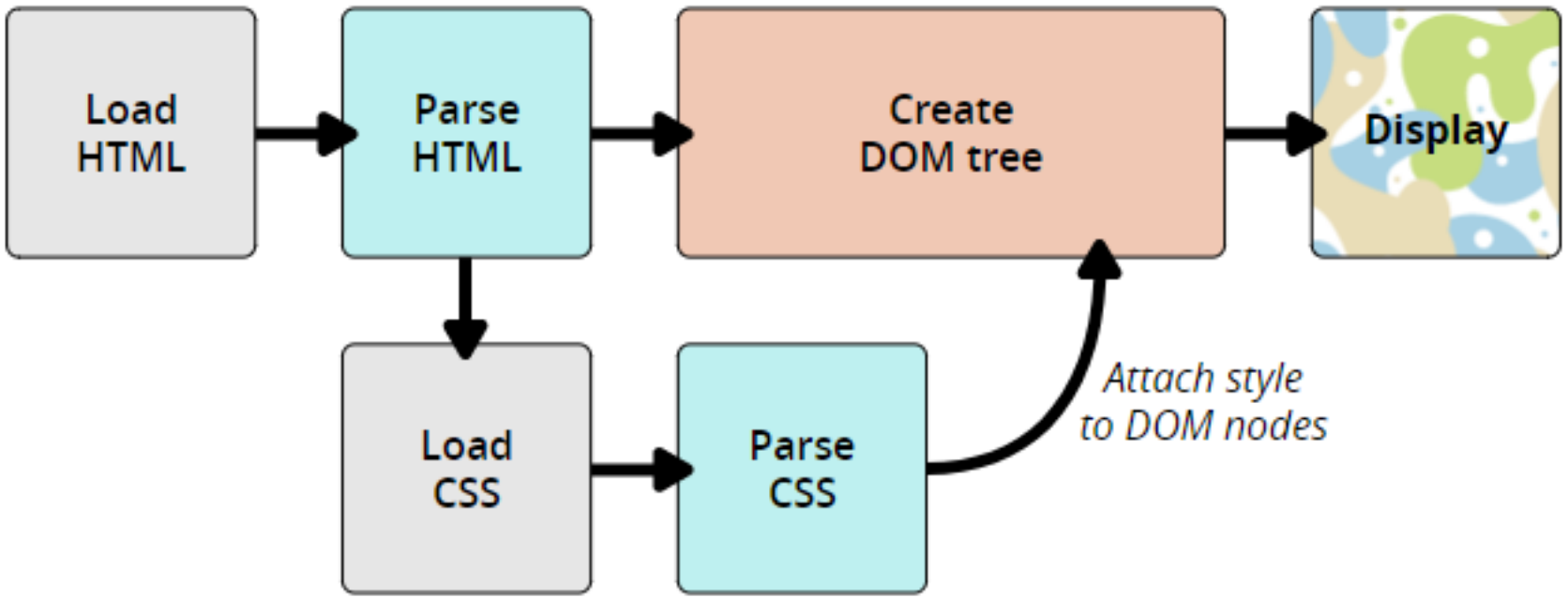
Bijgevolg erft het p-element de `font-family` en `font-size` over van zijn parent: het body-element

HO GENT

CSS verwerking door browsers

Cascading - inheritance

CSS : Hoe wordt CSS verwerkt door de browser



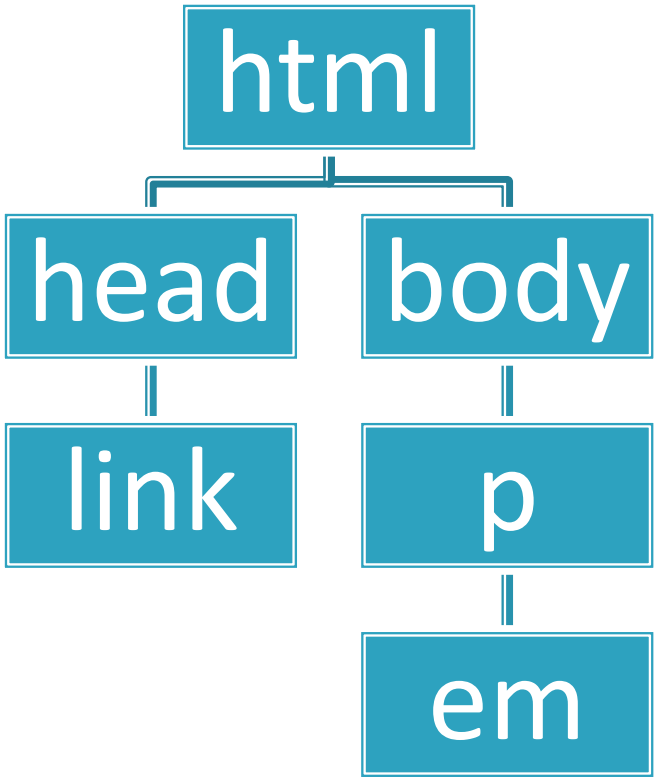
HO GENT

Create DOM tree

- Nadat de browser een HTML document geladen heeft, wordt een boomstructuur (DOM tree) van alle elementen gemaakt.
- Bijvoorbeeld

.html

```
<html>
   <head>
      <link rel="stylesheet" href="vb.css" />
   </head>
   <body>
      <p>
         Lorem <em> ipsum </em> dolor sit
         amet, consectetur adipiscing elit.
      </p>
    </body>
</html>
```

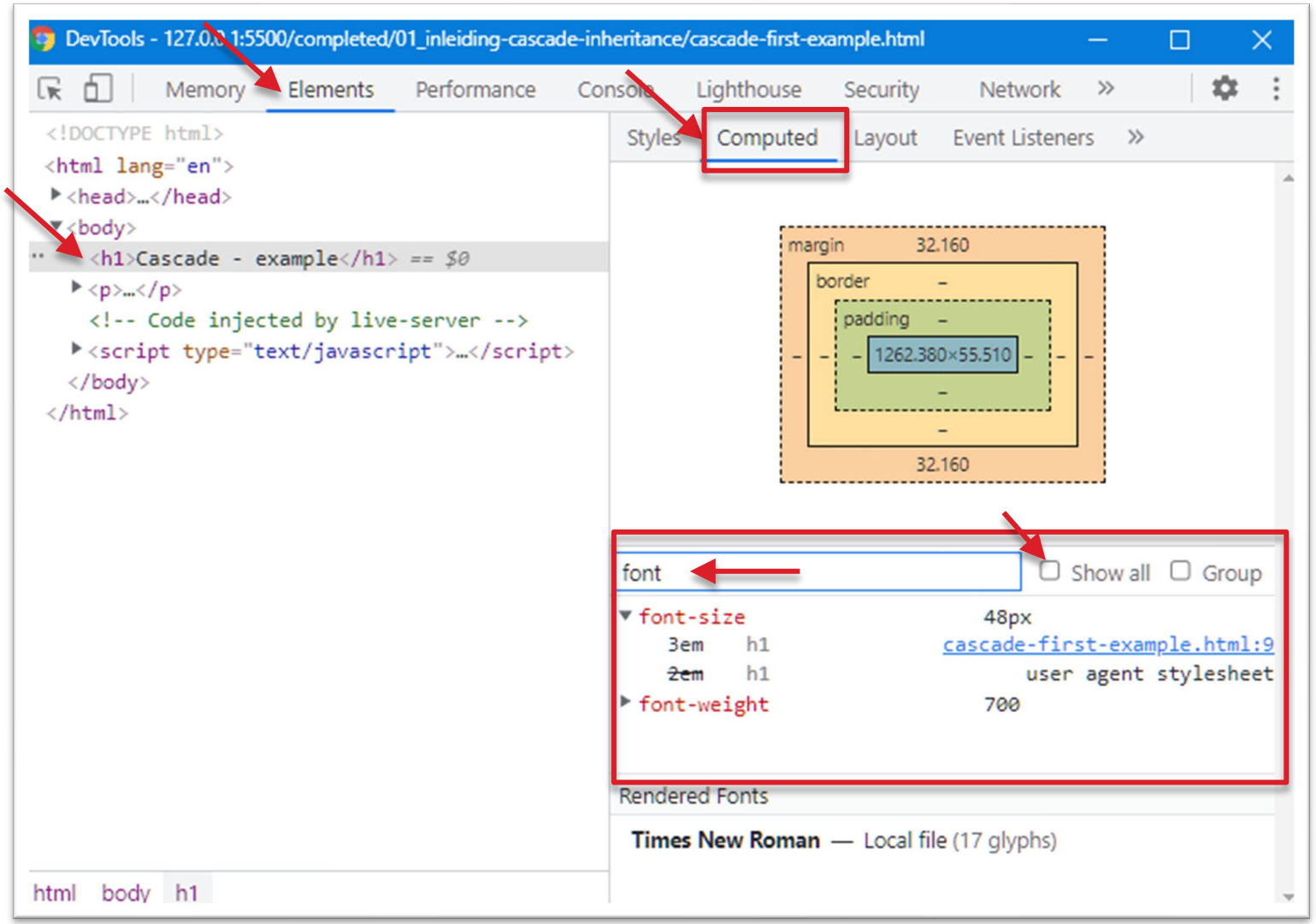


HO GENT

Hoe werkt CSS?

- Als de boomstructuur is opgebouwd, leest de browser alle stijlinformatie en berekent voor **elk HTML-element** in de boomstructuur **een waarde** voor **elke CSS-eigenschap**.
- De berekening van een waarde voor een eigenschap van een element gebeurt als volgt
  1. Werd er een waarde toegekend (in een stylesheet of inline) dan wordt die waarde gebruikt.
     Werd er meerdere keren een waarde toegekend. De cascade (zie verderop) bepaalt de waarde die wordt toegepast.
     Hierbij hebben inline declarations (style-attribuut in HTML) met dezelfde **importance** (zie verderop) altijd voorrang op declarations in een stylesheet.
  2. Indien geen waarde wordt toegekend, wordt nagegaan of er een waarde wordt overgeërfd (**inherited**)
  3. Zoniet, dan wordt de **initial value** voor deze eigenschap gebruikt.

Developer Tools - Computed value



HO GENT

De ‘Cascade’:

Origin, importance, Specificity, Source order

Voorbeeld Cascade: origin, importance

- Open **02-cascade/1-site-example/**

Cascade: origin, importance

- CSS stijlen kunnen afkomstig zijn van 3 verschillende stylesheets.
  - User agent stylesheet (User-Agent Origin)
  - User stylesheets (User Origin)
  - Author stylesheets (Author Origin)

- Daarnaast hebben we ook nog de inline styles (gekoppeld aan een HTML-element via het style-attribuut) die (standaard) voorrang hebben op alle andere styles.

```
<body style="font-family: serif;">
```

CSS: User agent stylesheet

- User agent stylesheet (Browser stylesheet)
  - Browsers hebben een default stylesheet.
  - Deze bevat de standaardstijlen die de browser toepast op een pagina. Deze kunnen worden overschreven in een user of author stylesheet.

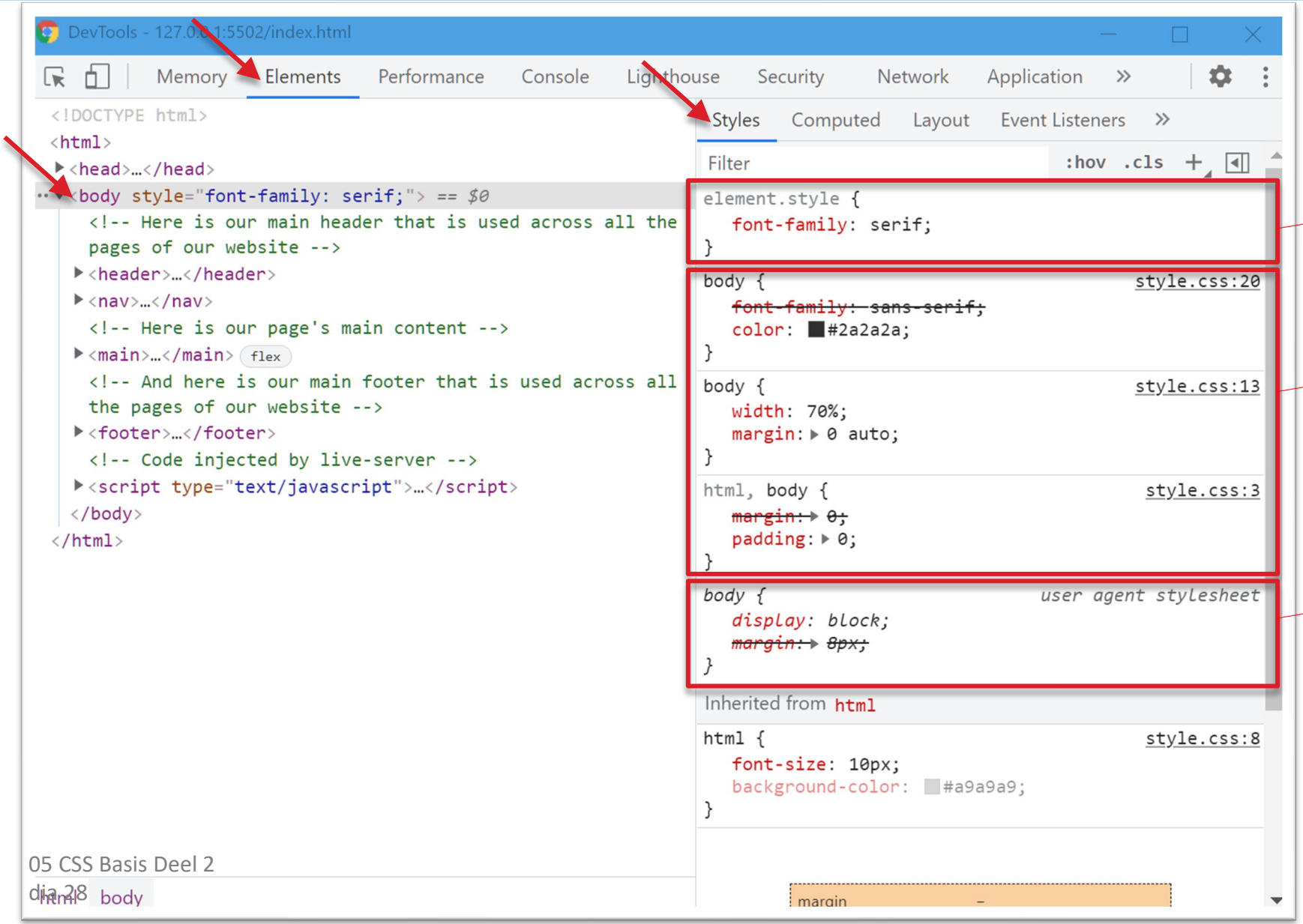
CSS: User stylesheet

- User stylesheets
  - Een bezoeker kan ook een eigen stylesheet aan de browser toevoegen omwille van user **accessibility** (grotere fonts, kleuren,…).
    Maar alhoewel dit opgenomen is in de beschrijving van de ‘Cascade’ wordt dit weinig gebruikt en gebeurt het aanpassen van webpagina’s omwille van user accessibility meestal via Browser extensies.

CSS: Author stylesheets

- Author stylesheets
  - Dit zijn de stijlen die wij als web developers instellen. Dit is onze CSS-code. Deze stijlen kunnen zich in één of meerdere .css-bestanden bevinden of in het <style>-element in de webpagina zelf.

Developer Tools: user agent stylesheet, author stylesheet



inline styles

author stylesheet

user agent stylesheet (browser stylesheet)

HO GENT

Voorbeeld Cascade: Importance

- Open **02-cascade/2-importance.html**

2-importance.html

- De prioriteit van een declaration kan verhoogd worden door er **!important** achter te plaatsen.
- Een **!important** declaratie heeft steeds de overhand op een gewone declaratie. Ook op een gewone declaratie in een HTML style-attribuut.

```
h1 {
  text-decoration: none !important;
}
```

De ‘Cascade’

- De cascade is het mechanisme waarbij een browser alle declaraties die op een element betrekking hebben een bepaalde prioriteit toekent en zo de uiteindelijke waarde bepaalt voor een eigenschap (intern zal de browser de declaraties hiervoor sorteren op prioriteit)
  - Eerst wordt gekeken naar de **origin** en de **importance**. Zo hebben (normale) declaraties in de ‘author stylesheet’ steeds voorrang op (normale) declaraties in de ‘user agent stylesheet’ (zie volgende dia.)
  - Hebben twee declaraties eenzelfde ‘origin’ en ‘importance’ dan wordt er gekeken welke van de bijhorende selectors de meest specifieke is. Met andere woorden welke de hoogste **specificity** heeft (zie verderop).
  - Als twee declaraties dezelfde ‘origin’, ‘importance’ en ‘specificity’ hebben dan wint de laatst gespecifieerde.

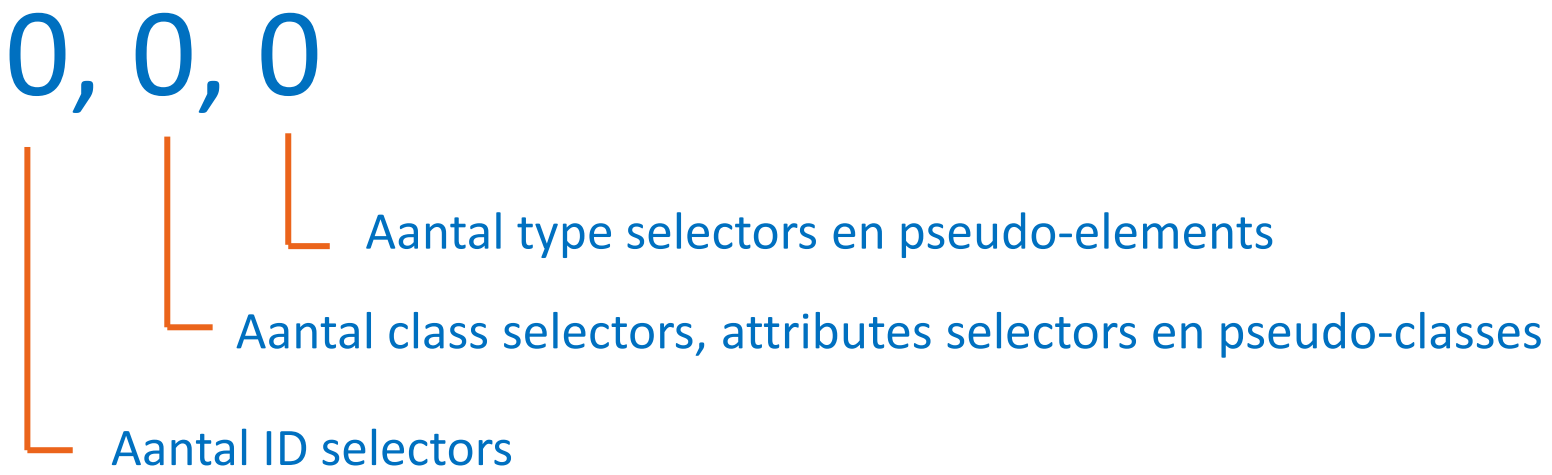
Cascade: origin - importance

- De voorrangsregels voor origin - importance zijn, in dalende volgorde
  - Important user agent declarations
  - Important user declarations
  - Important author declarations
  - **Normal author declarations**
  - Normal user declarations
  - **Normal user agent declarations**

- **Normal author declarations** overschrijven dus **Normal user agent declarations**.

Cascade: specificity

- Hebben twee declaraties eenzelfde ‘origin’ en ‘importance’ dan wint de declaration met de selector met de hoogste **specificity**.
- Hoe wordt de specificity van een selector bepaald?
  - Wordt uitgedrukt met 3 getallen (hundreds, tens, ones)



  - De universal selector (*) wordt niet meegeteld bij het berekenen van de specificity en heeft specificity 0.

Meer info: https://www.w3.org/TR/selectors/#specificity-rules

Voorbeeld berekening specificity

- Beschouw de CSS selector `article > p.notes`
  De specificity van deze selector is:

Opmerking Combinators hebben geen invloed op de Specificity. De volgende CSS selectors hebben dus allemaal specificity 0,1,2.

```
article > p.notes
.notes article p
article.notes + p
```


HO GENT

Voorbeelden berekening specificity

Selector	Hundreds	Tens	Ones	Total specificity
h1	0	0	1	0,0,1
h1 + p::first-letter	0	0	3	0,0,3
li > a[href*="en-US"] > .inline-warning	0	2	2	0,2,2
#content	1	0	0	1,0,0

Voorbeeld Cascade: specificity

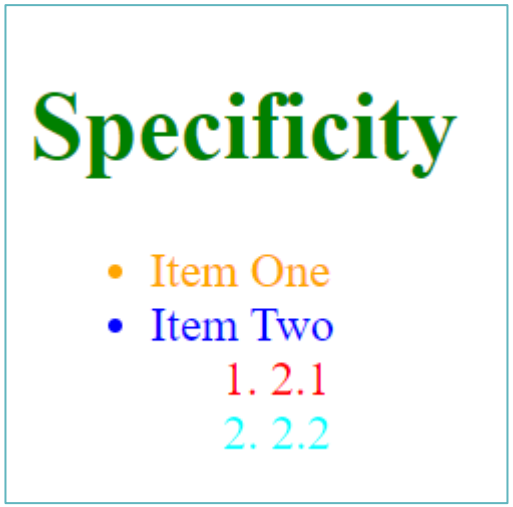
- Open **02-cascade/3-specificity.html**

Cascade: 3-specificity.html

- Als er meerdere conflicterende stijlen zijn op een element, bereken de specificiteit van elke stijl:

```css
/* specificity: 0,0,0 */
* {color: green;}

/* specificity: 0,0,1 */
li {color: aqua;}

/* specificity: 0,0,2 */
ul>li {color: orange;}

/* specificity: 0,1,1 */
li.red {color: red;}

/* specificity: 1,0,0 */
#content {color: blue;}
```

```html
<h1>Specificity</h1>
<ul>
  <li>Item One</li>
  <li id="content" class="red">
    Item Two
    <ol>
      <li class="red">2.1</li>
      <li>2.2</li>
    </ol>
  </li>
</ul>
```



Tip: als je in Visual Studio code hovert over een CSS selector wordt de specificiteit getoond.

HO GENT

Voorbeeld Cascade: source order

- Open **02-cascade/1-site-example/index.html**

Cascade: source order
(02-cascade/1-site-example/index.html)

- Als twee regels dezelfde **specificity** hebben dan wint de laatste:

```
p {
  color: green;
}
/* Alle p-elementen zijn grijs en niet groen */
p {
  color: grey;;
}
```

```
nav a {
  background-color: yellow;
}
/* De achtergrondkleur van alle a-elementen
   is wit en niet geel */
body a {
  background-color: white;
}
```

Inheritance - overerving

Voorbeeld Inheritance

- Open **03-inheritance/inheritance.html**

Overerfbare eigenschappen

- Werd er geen waarde opgegeven in de cascade, dan wordt er gekeken of de eigenschap overerfbaar is. Indien dit het geval is, dan wordt de waarde overgeërfd (**inherited**) van het parent element.

```
p {
  color: blue;
}
```

```
<p>Lorem <em> ipsum </em> dolor sit amet, consectetur adipiscing elit.</p>
```

`p` is het parent element van `em`. Dus als we een stijl `color: blue;` toekennen aan `p`, dan zal `em` deze stijl overerven en ook `color: blue;` hebben

Welke properties worden overgeërfd?

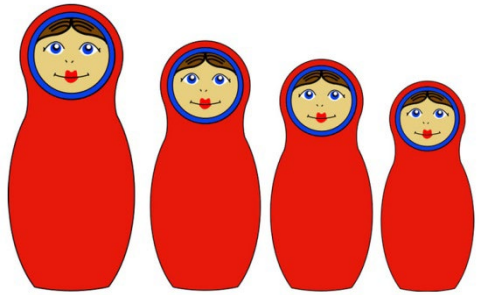
- Tekst gerelateerde properties:
  color, font-family, font-size, font-style, text-align, text-transform, ….
- List properties:
  list-style-image, list-style-position, list-style-type, list-style

Inheritance – computed value

- Via inheritance propageren de waarden van eigenschappen van parent elements naar hun children. Let echter op: de gepropageerde waarde is niet de waarde uit de declaratie maar de berekende waarde (**computed value**).
- Zo wordt in het voorbeeld hiernaast de **berekende** waarde (**in pixels**) van font-size overgeërfd in plaats van de waarde **0.8em**.
- **Stel** dat de property font-size gewoon geërfd werd, dan zou dat bijvoorbeeld het volgende effect geven: de font-size van het element zou dan 80% zijn van de font-size van het <p> element.

```
<p>Lorem <em> ipsum </em> dolor sit amet,
consectetur adipiscing elit.</p>
```

```
p {
  font-size: 0.8em;
}
```



Lorem *ipsum* dolor sit amet, consectetur adipiscing elit.

Lorem *ipsum* dolor sit amet, consectetur adipiscing elit.

Stel dat het - element de font-size: 0.8em had geërfd

HO GENT

Overerving eigenschappen

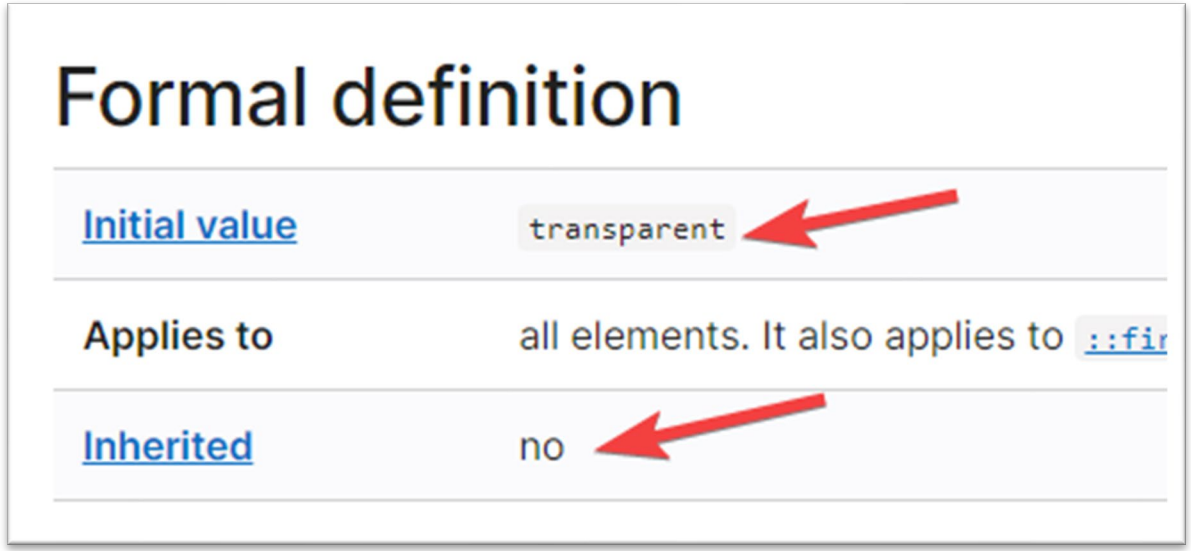
- Plaats overerfbare eigenschappen die voor de volledige pagina gelden bij body of in een wrapper div (omsluit alle andere elementen van de pagina).
- Als er voor een HTML-element geen expliciete waarde ingesteld is voor een CSS-property, dan erft het element de waarde van zijn dichtstbijzijnde ouder in de DOM-tree.
- Men kan steeds een niet overerfbare eigenschap toch doen overerven door middel van de value: inherit

Voorbeeld: de ‘initial value’

- Open **03-inheritance/initial-value.html**

Wat met niet overerfbare eigenschappen?

- Indien er voor een niet overerfbare eigenschap geen expliciete stijl is ingesteld dan wordt de **initial value** gebruikt.
- Voor elke CSS eigenschap kan je op MDN terugvinden of deze al dan niet overgeërfd wordt, alsook zijn **initial value**.
- Voorbeeld: MDN – background-color



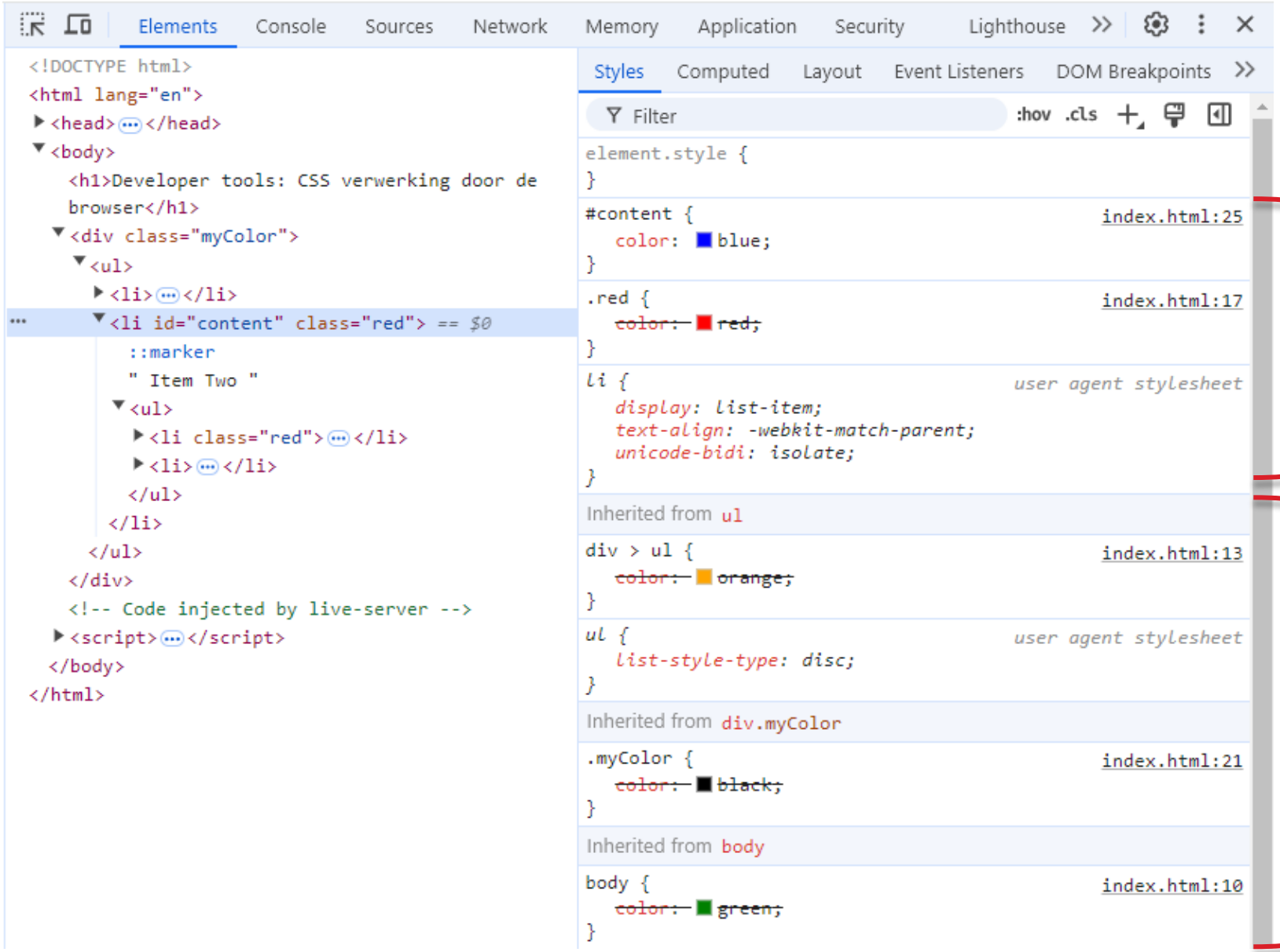
HO GENT

Developer Tools

Voorbeeld: Developer Tools

- Open **04-developer-tools/index.html**

Developer Tools – CSS verwerking door de browser



Dalende prioriteit in de ‘cascade’

Dalende prioriteit ‘inheritance’

HO GENT

CSS values and units

CSS values and units

- Elke CSS-eigenschap heeft een **value type** dat de mogelijke waarden voor de eigenschap bepaalt.
- **Value types** worden meestal genoteerd met punthaken om ze te onderscheiden van CSS-eigenschappen (color property versus <color> value type).
- Een veel voorkomend **value type** is <length>. Lengtes kunnen bij verschillende CSS properties gebruikt worden:

```
font-size: 0.8em;
margin: 0.8em;
width: 0.8em;
```

```
font-size: 10px;
margin: 10px;
width: 10px;
```

  Merk op dat er geen spatie staat tussen het getal en de eenheid.
- Voor meer info zie MDN CSS values and units.

Browser reset/normalize.css

Browser resets/normalize files

- De HTML-specificatie geeft suggesties aan Browser creators in verband met de Browser style sheet, maar legt geen verplichte regels op. Verschillende Browsers kunnen en hebben dus licht afwijkende Browser stylesheets (user agent stylesheets).
  Meer info HTML standard - Rendering
- Vroeger maakte men om dit probleem op te lossen dikwijls gebruik van de reset.css van Eric Meyer. Vandaag gebruikt men hiervoor eerder de normalize.css van Nicolas Gallagher, die minder stijlinformatie verwijdert.
- Vroeger waren er grote verschillen, maar vandaag zijn de verschillen miniem zodat het gebruik van een reset/normalize file niet altijd nodig is.

Toevoegen reset.css/normalize.css

- Als je gebruik wil maken van een reset.css/normalize.css
  - voeg dan een verwijzing toe naar deze css-file in elk van je html pagina's (als eerste link).

```
<head>
  …
  <title>….</title>
  <link rel="stylesheet" href="css/normalize.css" />
  <link rel="stylesheet" href="css/site.css" />
</head>
```

  - of gebruik een @import in je CSS-code

```
@import url("normalize.css");
```

Web fonts

Voorbeeld: Web fonts

- Open **05-web-fonts/1-web-fonts.html**
  Gebruik de info op de volgende slide om de h1- en de h2-elementen in deze webpagina weer te geven in het lettertype **AlexBrush**.
  De font-file voor dit lettertype bevindt zich in de map **fonts**.

Voorbeeld: Web fonts

- Vroeger kon de browser enkel de lettertypes gebruiken die op de computer van de gebruiker geïnstalleerd waren. Maar nu kunnen browsers ook fonts laden uit een font file (web fonts). Web fonts zijn dus lettertypes die niet standaard op iemands computer staan maar die vanaf een server worden geladen. Dit gebeurt via het @font-face CSS statement.

```
@font-face {
  font-family: 'my font';
  src: url(../fonts/AlexBrush-Regular.ttf);
}

h1, h2 {
  font-family: 'my font', cursive;
}
```

Web fonts - bestandsformaten

- De meeste browsers ondersteunen de volgende bestandsformaten:
  - .TTF (TrueType)
  - .OTF (OpenType)
  - .WOFF (Web Open Font Format)
  - .WOFF2 (Web Open Font Format 2)
- WOFF en WOFF2 zijn formaten die speciaal ontwikkeld werden voor het web.
- WOFF fonts zijn eigenlijk een gecomprimeerde versie van True Type (TTF) of Open Type fonts(OTF) => WOFF fonts zijn doorgaans kleiner en vlugger te downloaden
  - Wordt ondersteund door IE, Edge, Firefox, Chrome, Safari, Opera, iOS Safari (versie 5 en hoger), Android-browser vanaf 4.4

Waar vind je Web fonts?

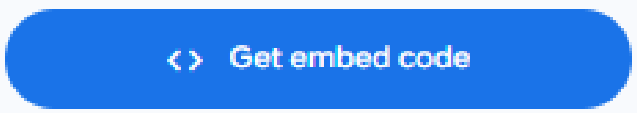
- Veel content op het web is auteursrechtelijk beschermd. Je mag bijvoorbeeld niet zomaar teksten of afbeeldingen van een andere website overnemen in je eigen website.
- Ook Web fonts zijn auteursrechtelijk beschermd. Je mag dus niet zomaar web fonts van een andere website gebruiken in je eigen website.
- Er bestaat een uitgebreid assortiment van gratis en betalende fonts:
  - **Font Squirrel** (https://www.fontsquirrel.com/): Gratis.
  - **Google fonts** (https://fonts.google.com/): Gratis. Bovendien wordt hier standaard verwezen naar de font files op de google servers en is het bijgevolg niet nodig om de font files in je website op te nemen.
  - **Adobe Fonts** (https://fonts.adobe.com/fonts): als je de fonts van Adobe wilt gebruiken moet je een abonnement nemen.
  - …

Voorbeeld: een google font gebruiken

- Gebruik de info op de volgende twee slides om de webpagina **05-web-fonts/2-google-fonts.html** weer te geven in het Google font **PT Sans**.

Google fonts (voorbeeld)

- Surf naar http://fonts.google.com
- Zoek het font **PT Sans** en klik erop om het te selecteren.
- Klik op vervolgens op 

- Klik op 

- Selecteer enkel het normale (Regular 400) en het vette lettertype (Bold 700) lettertype .
  (p-elementen worden standaard normaal weergegeven en h1- en h2-elementen worden standaard bold weergegeven).
- Kopieer de code voor het importeren en het gebruiken van de lettertypes naar Visual Studio Code => meer info vind je op de volgende dia.

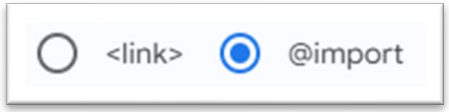
HO GENT

Google fonts (voorbeeld)

- ‘embedden’ van de fonts in je website.
  - De standaard manier is via het **<link>**-element in de <head> van je webpagina.
    Dit linkt naar een door Google gegenereerd CSS-bestand met @font-face regels.

    Opmerkingen

    - Google Fonts voegt standaard twee link-elementen met rel="preconnect" toe. Deze zijn niet noodzakelijk, maar kunnen het downloaden van de fonts wel versnellen. Zie ook MDN – preconnect.
  - Er is een tweede manier om het font te ‘embedden’ in je website namelijk met een CSS **@import** statement.



- De CSS-code om het lettertype te gebruiken is:

```
font-family: 'PT Sans', sans-serif;
```


HO GENT

Font pairing

- **Font Pairing**
  Een “font pair” is een verzameling van twee of meer lettertypes die je zal gebruiken in je website. Meestal bevat een “font pair” een lettertype voor de koppen en een lettertype voor de tekst.
  Een goede font pairing bepaalt de “visual hierarchy” en verbetert de leesbaarheid.
  - Inspiratie?
    - https://www.w3schools.com/css/css_font_pairings.asp
    - http://fontpair.co/ **> pairing**

- Voor meer extra info over web fonts zie MDN – Web fonts.